Introduction to Grid Computing

The Globus Project™
Argonne National Laboratory
USC Information Sciences Institute
http://www.globus.org/

Outline

- Introduction to Grid Computing
- Some Definitions
- Grid Architecture
- The Programming Problem
- The Globus Toolkit™
  - Introduction, Security, Resource Management, Information Services, Data Management
- Related work
- Futures and Conclusions

The Grid Problem

- Flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resource

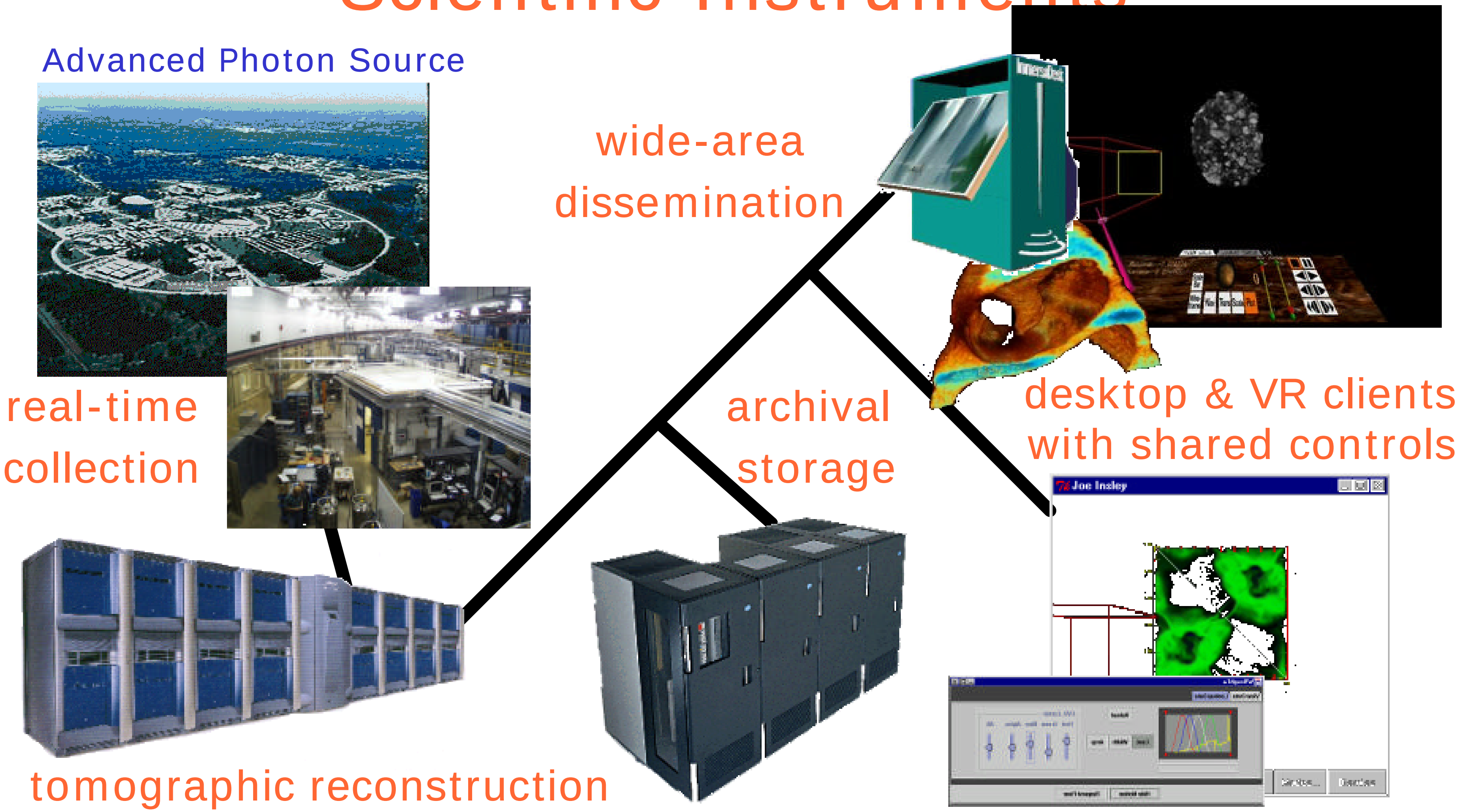
  From “The Anatomy of the Grid: Enabling Scalable Virtual Organizations”

- Enable communities (“virtual organizations”) to share geographically distributed resources as they pursue common goals -- *assuming the absence* of…
  - central location,
  - central control,
  - omniscience,
  - existing trust relationships.

Elements of the Problem

- Resource sharing
  - Computers, storage, sensors, networks, ...
  - Sharing always conditional: issues of trust, policy, negotiation, payment, ...
- Coordinated problem solving
  - Beyond client-server: distributed data analysis, computation, collaboration, ...
- Dynamic, multi-institutional virtual orgs
  - Community overlays on classic org structures
  - Large or small, static or dynamic

Online Access to Scientific Instruments



DOE X-ray grand challenge: ANL, USC/ISI, NIST, U.Chicago

Data Grids for High Energy Physics

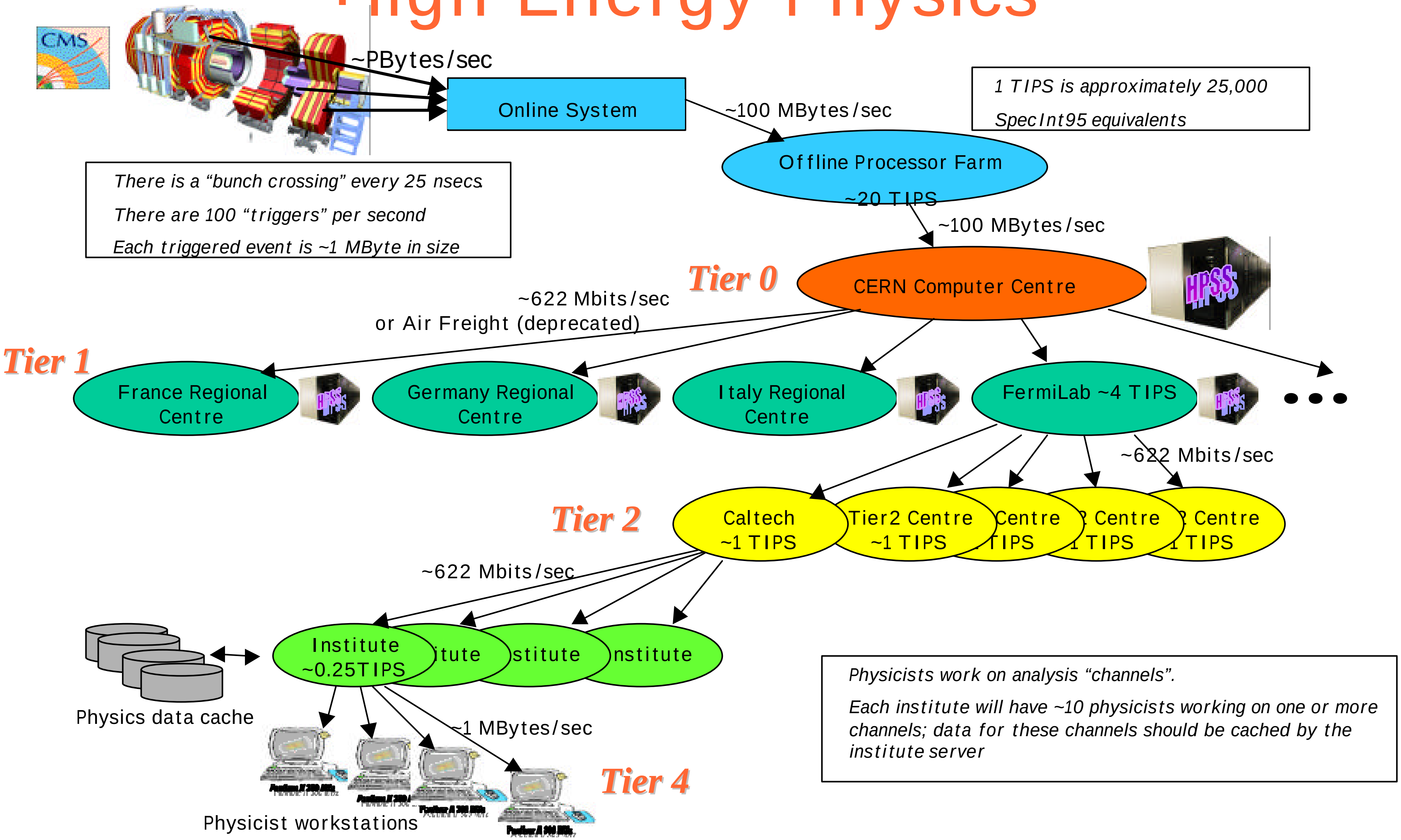
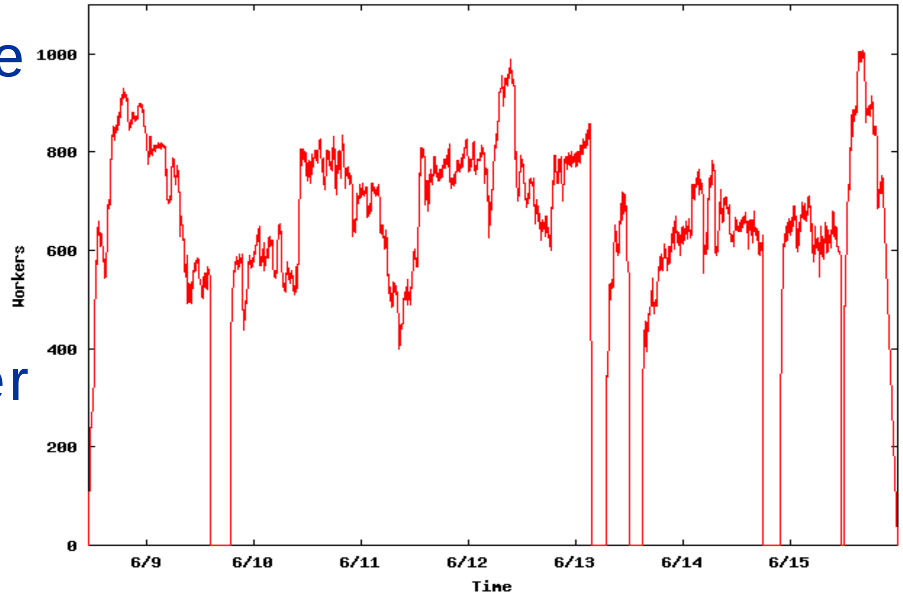


Image courtesy Harvey Newman, Caltech

Mathematicians Solve NUG30

- Looking for the solution to the NUG30 quadratic assignment problem
- An informal collaboration of mathematicians and computer scientists
- Condor-G delivered 3.46E8 CPU seconds in 7 days (peak 1009 processors) in U.S. and Italy (8 sites)

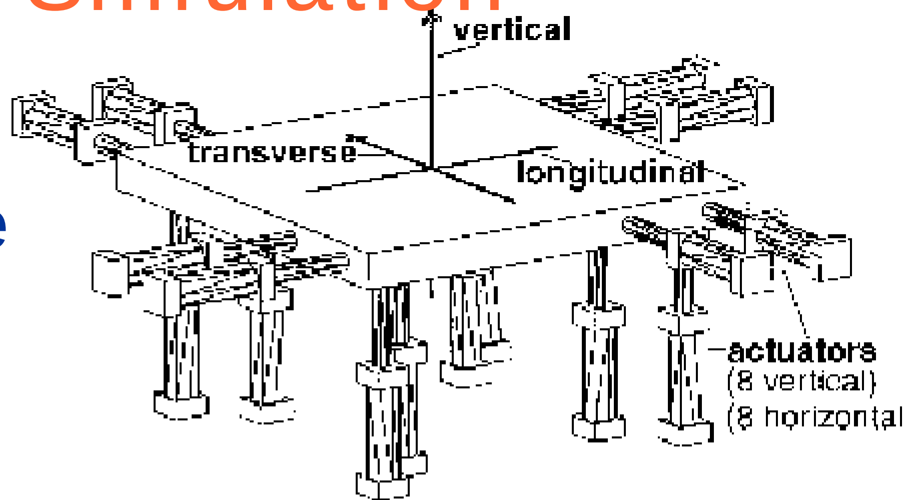


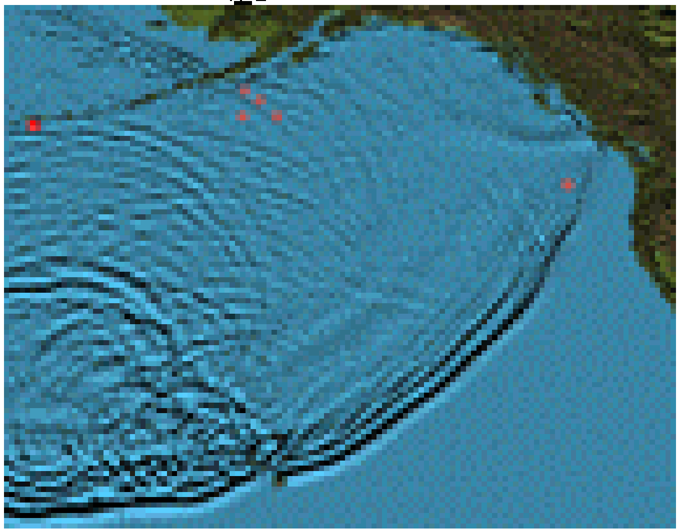
14,5,28,24,1,3,16,15,
10,9,21,2,4,29,25,22,
13,26,17,30,6,20,19,
8,18,7,27,12,11,23

MetaNEOS: Argonne, Iowa, Northwestern, Wisconsin

Network for Earthquake Engineering Simulation

- NEESgrid: US national infrastructure to couple earthquake engineers with experimental facilities, databases, computers, & each other
- On-demand access to experiments, data streams, computing, archives, collaboration





NEESgrid: Argonne, Michigan, NCSA, UIUC, USC

Home Computers Evaluate AIDS Drugs

- Community =
  - 1000s of home computer users
  - Philanthropic computing vendor (Entropia)
  - Research group (Scripps)
- Common goal= advance AIDS resear

Broader Context

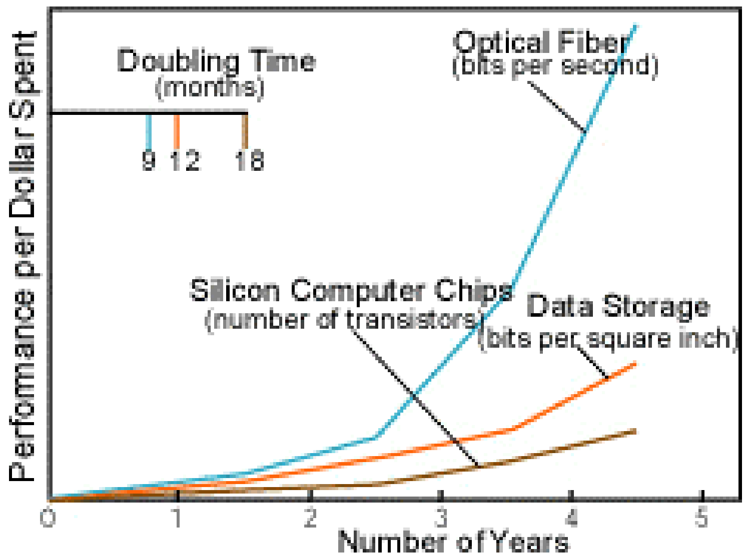
- "Grid Computing" has much in common with major industrial thrusts
  - Business-to-business, Peer-to-peer, Application Service Providers, Storage Service Providers, Distributed Computing, Internet Computing…
- Sharing issues not adequately addressed by existing technologies
  - Complicated requirements: "run program X at site Y subject to community policy P, providing access to data at Z according to policy Q"
  - High performance: unique demands of advanced & high-performance systems

Why Now?

- Moore's law improvements in computing produce highly functional endsystems
- The Internet and burgeoning wired and wireless provide universal connectivity
- Changing modes of working and problem solving emphasize teamwork, computation
- Network exponentials produce dramatic changes in geometry and geography

Network Exponentials

- Network vs. computer performance
  - Computer speed doubles every 18 months
  - Network speed doubles every 9 months
  - Difference = order of magnitude per 5 years
- 1986 to 2000
  - Computers: x 500
  - Networks: x 340,000
- 2001 to 2010
  - Computers: x 60
  - Networks: x 4000



Moore's Law vs. storage improvements vs. optical improvements. Graph from **Scientific American** (Jan-2001) by Cleo Vilett, source Vined Khoslan, Kleiner, Caufield and Perkins.

The Globus Project™

Making Grid computing a reality

- Close collaboration with real Grid projects in science and industry
- Development and promotion of standard Grid protocols to enable interoperability and shared infrastructure
- Development and promotion of standard Grid software APIs and SDKs to enable portability and code sharing
- The Globus Toolkit™: Open source, reference software base for building grid infrastructure and applications
- Global Grid Forum: Development of standard protocols and APIs for Grid computing

Selected Major Grid Projects

Name	URL & Sponsors	Focus
Access Grid	www.mcs.anl.gov/FL/ accessgrid; DOE, NSF	Create & deploy group collaboration systems using commodity technologies
BlueGrid *New*	IBM	Grid testbed linking IBM laboratories
DISCOM	www.cs.sandia.gov/ discom DOE Defense Programs	Create operational Grid providing access to resources at three U.S. DOE weapons laboratories
DOE Science Grid *New*	sciencegrid.org DOE Office of Science	Create operational Grid providing access to resources & applications at U.S. DOE science laboratories & partner universities
Earth System Grid (ESG)	earthsystemgrid.org DOE Office of Science	Delivery and analysis of large climate model datasets for the climate research community
European Union (EU) DataGrid	eu-datagrid.org European Union	Create & apply an operational grid for applications in high energy physics, environmental science, bioinformatics

Selected Major Grid Projects

Name	URL/Sponso	Focus
EuroGrid, Grid Interoperability (GRIP) *New*	eurogrid.org European Union	Create tech for remote access to supercomp resources & simulation codes; in GRIP, integrate with Globus Toolkit™
Fusion Collaboratory *New*	fusiongrid.org DOE Off. Science	Create a national computational collaboratory for fusion research
Globus Project™	globus.org DARPA, DOE, NSF, NASA, Msoft	Research on Grid technologies; development and support of Globus Toolkit™; application and deployment
GridLab *New*	gridlab.org European Union	Grid technologies and applications
GridPP *New*	gridpp.ac.uk U.K. eScience	Create & apply an operational grid within the U.K. for particle physics research
Grid Research Integration Dev. & Support Center *New*	grids-center.org NSF	Integration, deployment, support of the NSF Middleware Infrastructure for research & education

Selected Major Grid Projects

Name	URL/Sponsor	Focus
Grid Application Dev. Software	hipersoft.rice.edu/grads; NSF	Research into program development technologies for Grid applications
Grid Physics Network	griphyn.org NSF	Technology R&D for data analysis in physics expts: ATLAS, CMS, LIGO, SDSS
Information Power Grid	ipg.nasa.gov NASA	Create and apply a production Grid for aerosciences and other NASA missions
International Virtual Data Grid Laboratory *New*	ivdgl.org NSF	Create international Data Grid to enable large-scale experimentation on Grid technologies & applications
Network for Earthquake Eng. Simulation Grid *New*	neesgrid.org NSF	Create and apply a production Grid for earthquake engineering
Particle Physics Data Grid	ppdg.net DOE Science	Create and apply production Grids for data analysis in high energy and nuclear physics experiments

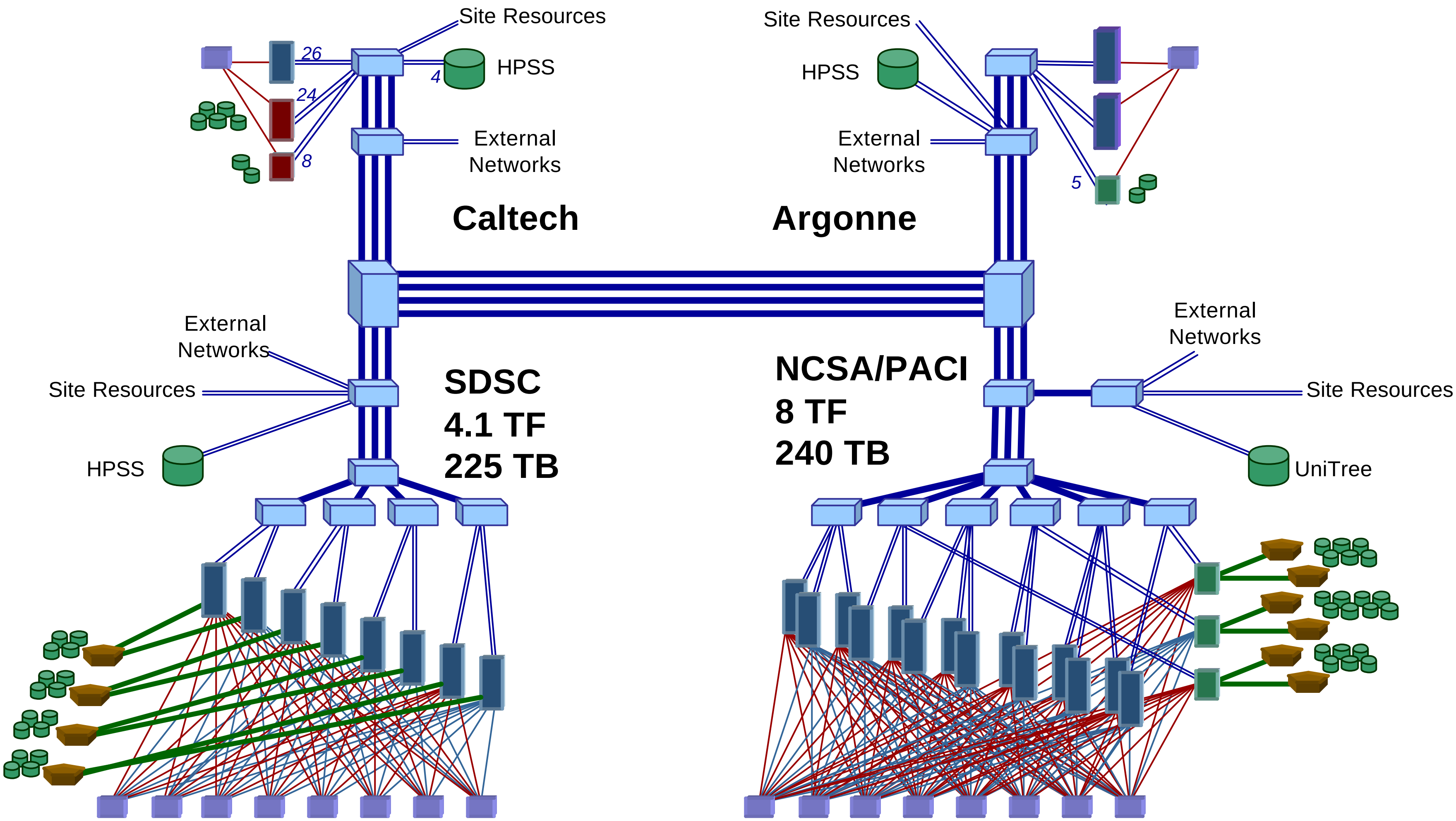
Selected Major Grid Projects

Name	URL/Sponsor	Focus
TeraGrid *New*	teragrid.org NSF	U.S. science infrastructure linking four major resource sites at 40 Gb/s
UK Grid Support Center *New*	grid-support.ac.uk U.K. eScience	Support center for Grid projects within the U.K.
Unicore	BMBFT	Technologies for remote access to supercomputers

Also many technology R&D projects:
e.g., Condor, NetSolve, Ninf, NWS

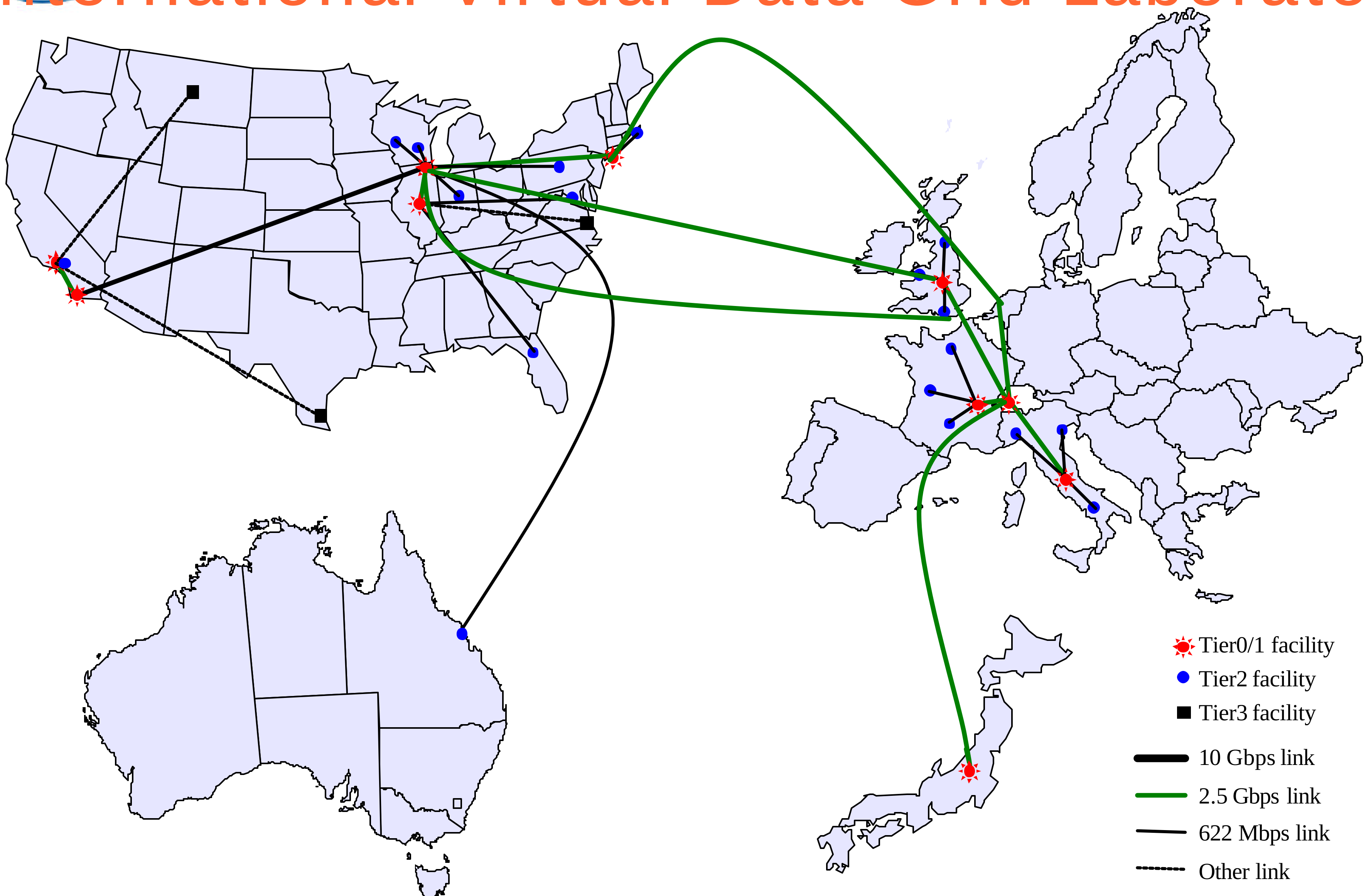
See also www.gridforum.org

The 13.6 TF TeraGrid: Computing at 40 Gb/s



TeraGrid/DTF: NCSA, SDSC, Caltech, Argonne   www.teragrid.org

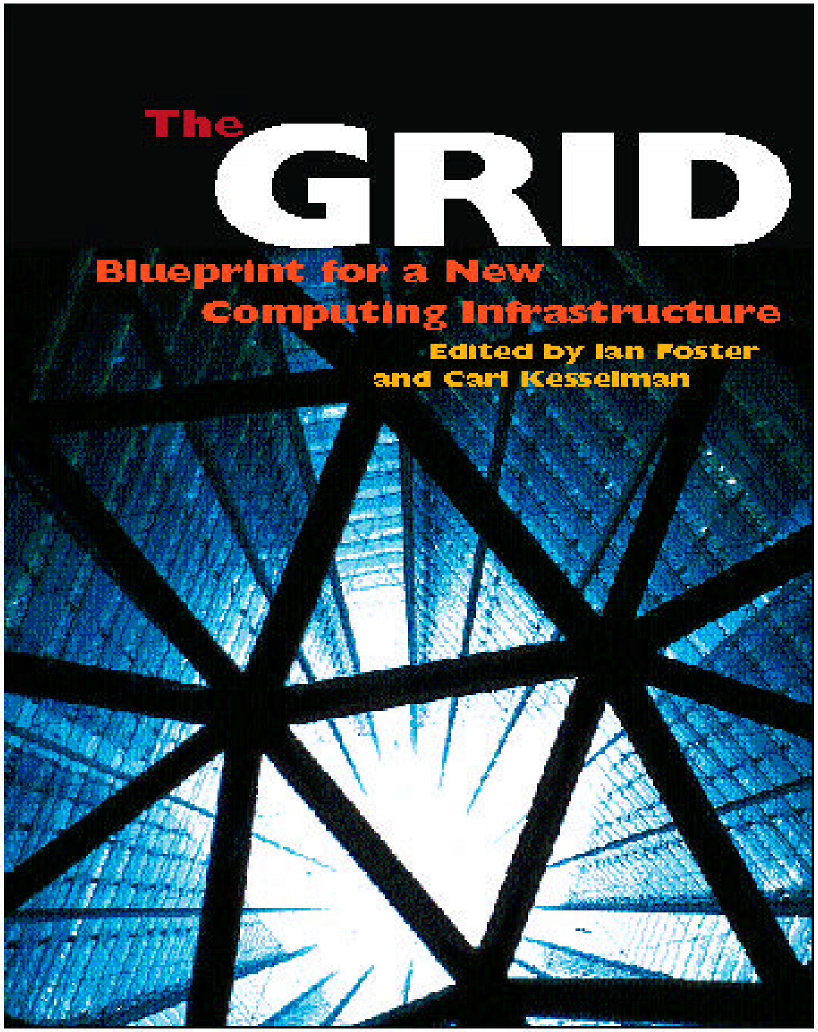
iVDGL: International Virtual Data Grid Laboratory



U.S. PIs: Avery, Foster, Gardner, Newman, Szalay   www.ivdgl.org

For More Information

- Globus Project™
  – www.globus.org
- Grid Forum
  – www.gridforum.org
- Book (Morgan Kaufman)
  – www.mkp.com/grids



Some Definitions

Some Important Definitions

- Resource
- Network protocol
- Network enabled service
- Application Programmer Interface (API)
- Software Development Kit (SDK)
- Syntax

- Not discussed, but important: policies

Resource

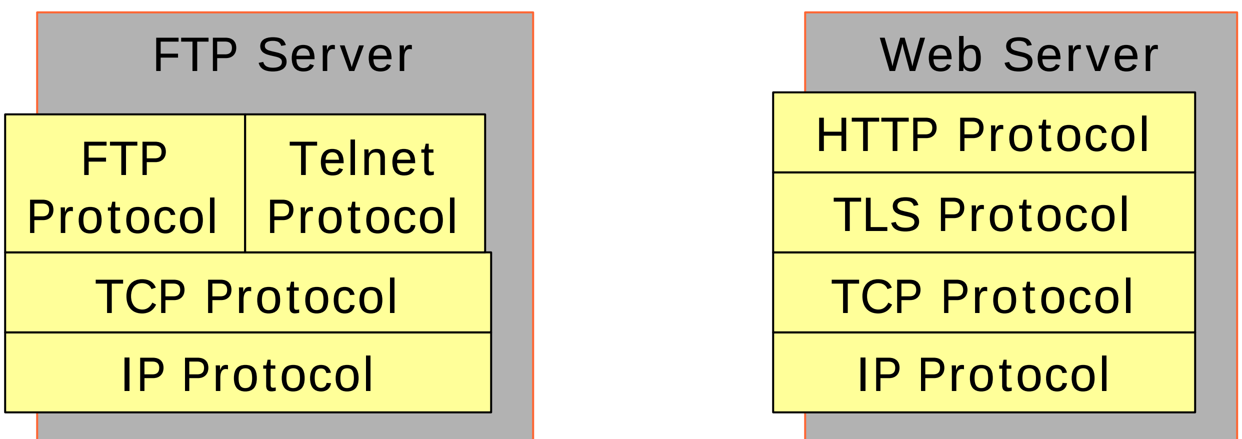
- An entity that is to be shared
  - E.g., computers, storage, data, software
- Does <u>not</u> have to be a physical entity
  - E.g., Condor pool, distributed file system, ...
- Defined in terms of interfaces, not devices
  - E.g. scheduler such as LSF and PBS define a compute resource
  - Open/close/read/write define access to a distributed file system, e.g. NFS, AFS, DFS

Network Protocol

- A formal description of message formats and a set of rules for message exchange
  - Rules may define sequence of message exchanges
  - Protocol may define state-change in endpoint, e.g., file system state change
- Good protocols designed to do one thing
  - Protocols can be layered
- Examples of protocols
  - IP, TCP, TLS (was SSL), HTTP, Kerberos

Network Enabled Services

- Implementation of a protocol that defines a set of capabilities
  – Protocol defines interaction with service
  – All services require protocols
  – Not all protocols are used to provide services (e.g. IP, TLS)
- Examples: FTP and Web servers



Application Programming Interface

- A specification for a set of routines to facilitate application development
  - Refers to definition, not implementation
  - E.g., there are many implementations of MPI
- Spec often language-specific (or IDL)
  - Routine name, number, order and type of arguments; mapping to language constructs
  - Behavior or function of routine
- Examples
  - GSS API (security), MPI (message passing)

Software Development Kit

- A particular instantiation of an API
- SDK consists of libraries and tools
  - Provides implementation of API specification
- Can have multiple SDKs for an API
- Examples of SDKs
  - MPICH, Motif Widgets

Syntax

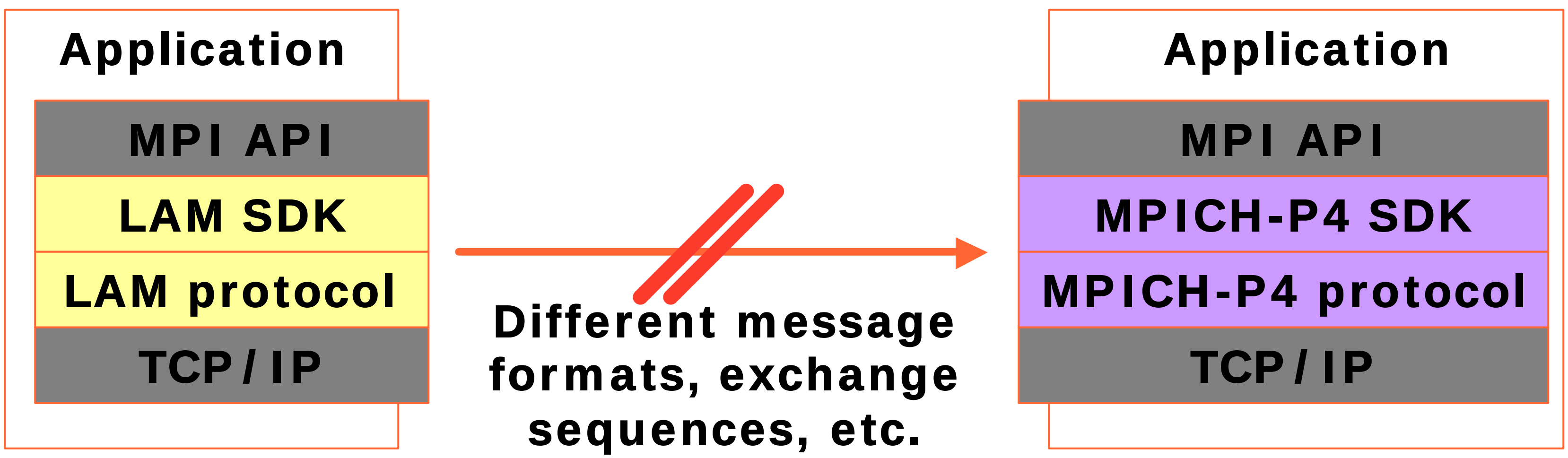
- Rules for encoding information, e.g.
  - XML, Condor ClassAds, Globus RSL
  - X.509 certificate format (RFC 2459)
  - Cryptographic Message Syntax (RFC 2630)
- Distinct from protocols
  - One syntax may be used by many protocols (e.g., XML); & useful for other purposes
- Syntaxes may be layered
  - E.g., Condor ClassAds -> XML -> ASCII
  - Important to understand layerings when comparing or evaluating syntaxes

A Protocol can have Multiple APIs

- TCP/IP APIs include BSD sockets, Winsock, System V streams, …
- The protocol provides <u>interoperability</u>: programs using different APIs can exchange information
- I don’t need to know remote user’s API

An API can have Multiple Protocols

- MPI provides portability: any correct program compiles & runs on a platform
- Does not provide interoperability: all processes must link against same SDK
  - E.g., MPICH and LAM versions of MPI



APIs and Protocols are Both Important

- Standard APIs/SDKs are important
  - They enable application *portability*
  - But w/o standard protocols, interoperability is hard (every SDK speaks every protocol?)
- Standard protocols are important
  - Enable cross-site *interoperability*
  - Enable shared infrastructure
  - But w/o standard APIs/SDKs, application portability is hard (different platforms access protocols in different ways)

the globus project™
www.globus.org

Grid Architecture

Why Discuss Architecture?

- Descriptive
  - Provide a common vocabulary for use when describing Grid systems
- Guidance
  - Identify key areas in which services are required
- Prescriptive
  - Define standard “Intergrid” protocols and APIs to facilitate creation of interoperable Grid systems and portable applications

One View of Requirements

- Identity & authentication
- Authorization & policy
- Resource discovery
- Resource characterization
- Resource allocation
- (Co-)reservation, workflow
- Distributed algorithms
- Remote data access
- High-speed data transfer
- Performance guarantees
- Monitoring
- Adaptation
- Intrusion detection
- Resource management
- Accounting & payment
- Fault management
- System evolution
- Etc.
- Etc.
- …

Another View: "Three Obstacles to Making Grid Computing Routine"

- New approaches to problem solving
  - Data Grids, distributed computing, peer-to-peer, collaboration grids, ...
- Structuring and writing programs
  - Abstractions, tools

Programming Problem

- Enabling resource sharing across distinct institutions

Systems Problem

  - Resource discovery, access, reservation, allocation; authentication, authorization, policy; communication; fault detection and notification; ...

Programming & Systems Problems

- The programming problem
  - Facilitate development of sophisticated apps
  - Facilitate code sharing
  - Requires <u>programming environments</u>
    - APIs, SDKs, tools
- The systems problem
  - Facilitate coordinated use of diverse resources
  - Facilitate infrastructure sharing
    - e.g., certificate authorities, information services
  - Requires <u>systems</u>
    - protocols, services

The Systems Problem: Resource Sharing Mechanisms That …

- Address security and policy concerns of resource owners and users
- Are flexible enough to deal with many resource types and sharing modalities
- Scale to large number of resources, many participants, many program components
- Operate efficiently when dealing with large amounts of data & computation

Aspects of the Systems Problem

- Need for interoperability when different groups want to share resources
  - Diverse components, policies, mechanisms
  - E.g., standard notions of identity, means of communication, resource descriptions
- Need for shared infrastructure services to avoid repeated development, installation
  - E.g., one port/service/protocol for remote access to computing, not one per tool/appln
  - E.g., Certificate Authorities: expensive to run
- A common need for protocols & services

Hence, a Protocol-Oriented View of Grid Architecture, that Emphasizes …

- Development of Grid protocols & services
  - Protocol-mediated access to remote resources
  - New services: e.g., resource brokering
  - “On the Grid” = speak Intergrid protocols
  - Mostly (extensions to) existing protocols
- Development of Grid APIs & SDKs
  - Interfaces to Grid protocols & services
  - Facilitate application development by supplying higher-level abstractions
- The (hugely successful) model is the Internet

the globus project
www.globus.org

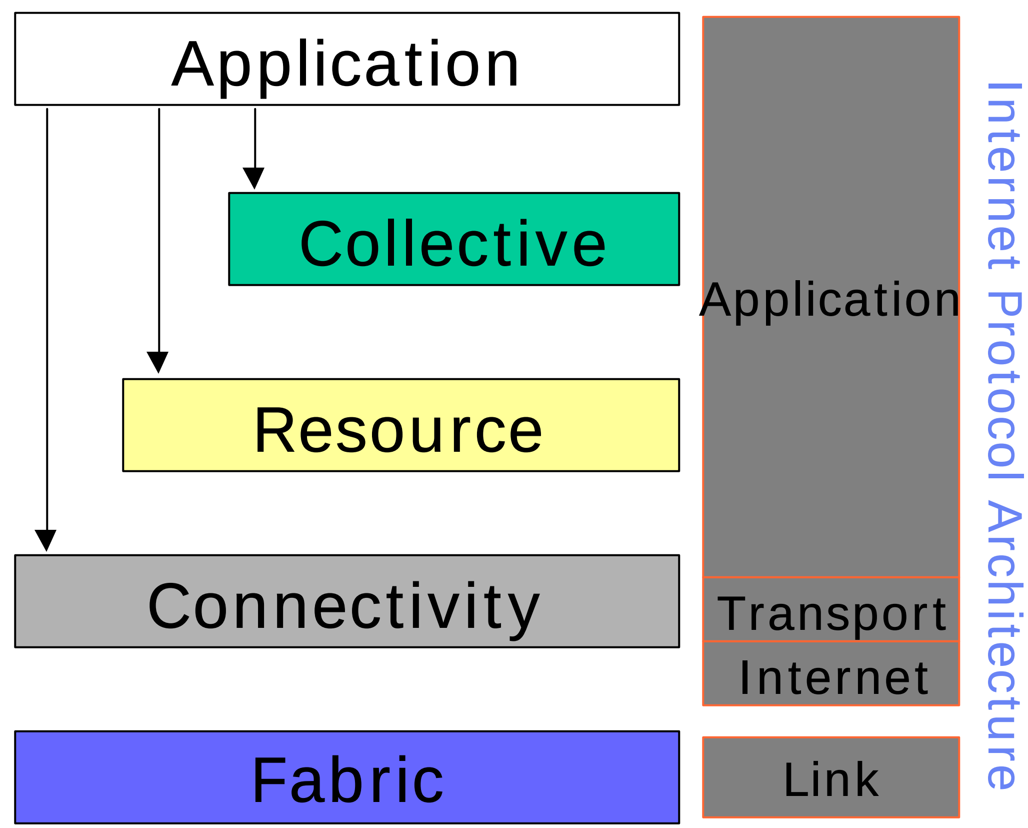
Layered Grid Architecture (By Analogy to Internet Architecture)

"Coordinating multiple resources": ubiquitous infrastructure services, app-specific distributed services

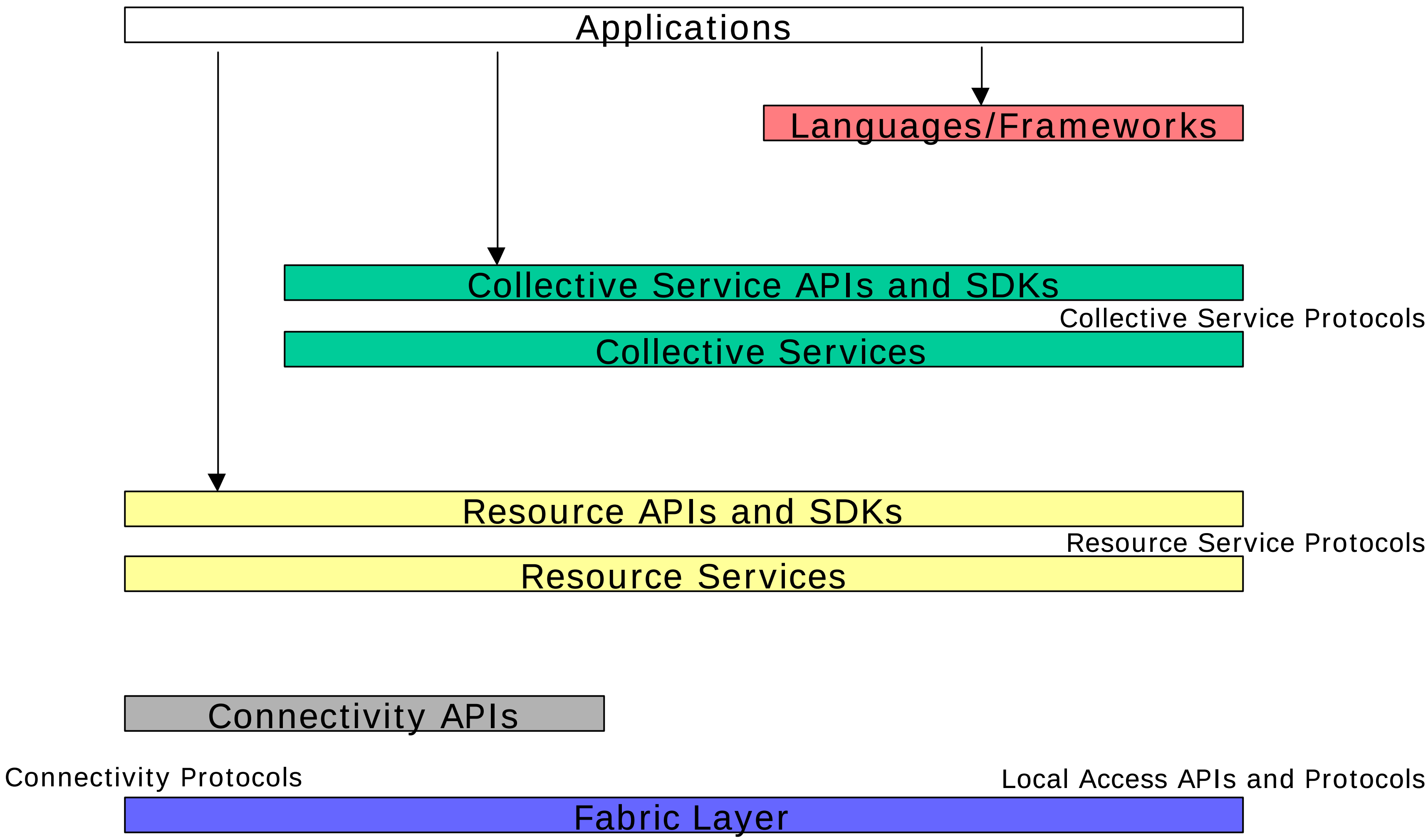
"Sharing single resources": negotiating access, controlling use

"Talking to things": communication (Internet protocols) & security

"Controlling things locally": Access to, & control of, resources



Protocols, Services, and APIs Occur at Each Level

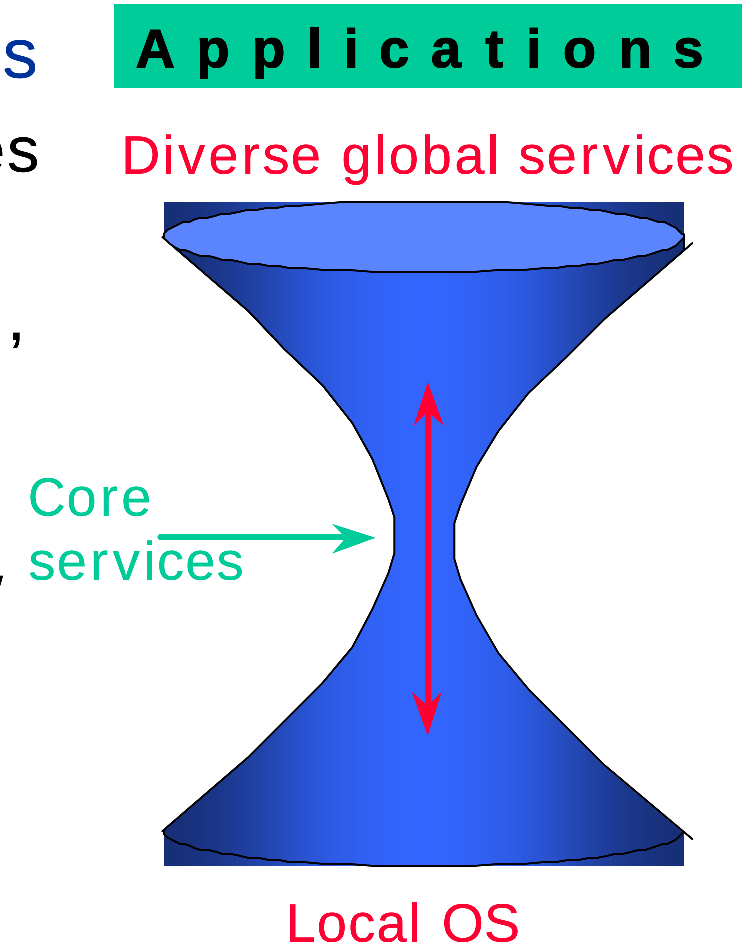


Important Points

- Built on Internet protocols & services
  - Communication, routing, name resolution, etc.
- “Layering” here is conceptual, does not imply constraints on who can call what
  - Protocols/services/APIs/SDKs will, ideally, be largely self-contained
  - Some things are fundamental: e.g., communication and security
  - But, advantageous for higher-level functions to use common lower-level functions

The Hourglass Model

- Focus on architecture issues
  - Propose set of core services as basic infrastructure
  - Use to construct high-level, domain-specific solutions
- Design principles
  - Keep participation cost low
  - Enable local control
  - Support for adaptation
  - “IP hourglass” model



Where Are We With Architecture?

- No “official” standards exist
- But:
  - Globus Toolkit™ has emerged as the de facto standard for several important Connectivity, Resource, and Collective protocols
  - GGF has an architecture working group
  - Technical specifications are being developed for architecture elements: e.g., security, data, resource management, information
  - Internet drafts submitted in security area

Fabric Layer Protocols & Services

- Just what you would expect: the diverse mix of resources that may be shared
  – Individual computers, Condor pools, file systems, archives, metadata catalogs, networks, sensors, etc., etc.
- Few constraints on low-level technology: connectivity and resource level protocols form the "neck in the hourglass"
- Defined by interfaces not physical characteristics

Connectivity Layer Protocols & Services

- Communication
  - Internet protocols: IP, DNS, routing, etc.
- Security: Grid Security Infrastructure (GSI)
  - Uniform authentication, authorization, and message protection mechanisms in multi-institutional setting
  - Single sign-on, delegation, identity mapping
  - Public key technology, SSL, X.509, GSS-API
  - Supporting infrastructure: Certificate Authorities, certificate & key management, …

GSI: www.gridforum.org/security/gsi

the globus project™
www.globus.org

Resource Layer Protocols & Services

- Grid Resource Allocation Management (GRAM)
  – Remote allocation, reservation, monitoring, control of compute resources
- GridFTP protocol (FTP extensions)
  – High-performance data access & transport
- Grid Resource Information Service (GRIS)
  – Access to structure & state information
- Others emerging: Catalog access, code repository access, accounting, etc.
- All built on connectivity layer: GSI & IP

GRAM, GridFTP, GRIS: www.globus.org

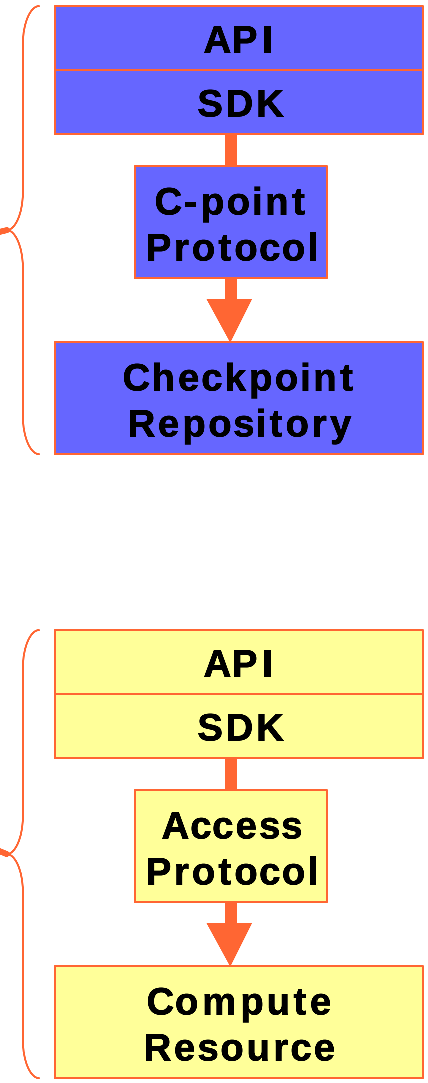
Collective Layer Protocols & Services

- Index servers aka metadirectory services
  - Custom views on dynamic resource collections assembled by a community
- Resource brokers (e.g., Condor Matchmaker)
  - Resource discovery and allocation
- Replica catalogs
- Replication services
- Co-reservation and co-allocation services
- Workflow management services
- Etc.

Condor: www.cs.wisc.edu/condor

Example: High-Throughput Computing System

App	High Throughput Computing System
Collective (App)	Dynamic checkpoint, job management, failover, staging
Collective (Generic)	Brokering, certificate authorities
Resource	Access to data, access to computers, access to network performance data
Connect	Communication, service discovery (DNS), authentication, authorization, delegation
Fabric	Storage systems, schedulers



Example: Data Grid Architecture

App	Discipline-Specific Data Grid Application
Collective (App)	Coherency control, replica selection, task management, virtual data catalog, virtual data code catalog, …
Collective (Generic)	Replica catalog, replica management, co-allocation, certificate authorities, metadata catalogs,
Resource	Access to data, access to computers, access to network performance data, …
Connect	Communication, service discovery (DNS), authentication, authorization, delegation
Fabric	Storage systems, clusters, networks, network caches, …

The Programming Problem

The Programming Problem

- But how do I develop robust, secure, long-lived, well-performing applications for dynamic, heterogeneous Grids?
- I need, presumably:
  - Abstractions and models to add to speed/robustness/etc. of development
  - Tools to ease application development and diagnose common problems
  - Code/tool sharing to allow reuse of code components developed by others

Grid Programming Technologies

- "Grid applications" are incredibly diverse (data, collaboration, computing, sensors, ...)
  - Seems unlikely there is one solution
- Most applications have been written "from scratch," with or without Grid services
- Application-specific libraries have been shown to provide significant benefits
- No new language, programming model, etc., has yet emerged that transforms things
  - But certainly still quite possible

Examples of Grid Programming Technologies

- MPICH-G2: Grid-enabled message passing
- CoG Kits, GridPort: Portal construction, based on N-tier architectures
- GDMP, Data Grid Tools, SRB: replica management, collection management
- Condor-G: workflow management
- Legion: object models for Grid computing
- Cactus: Grid-aware numerical solver framework
  - Note tremendous variety, application focus

MPICH-G2: A Grid-Enabled MPI

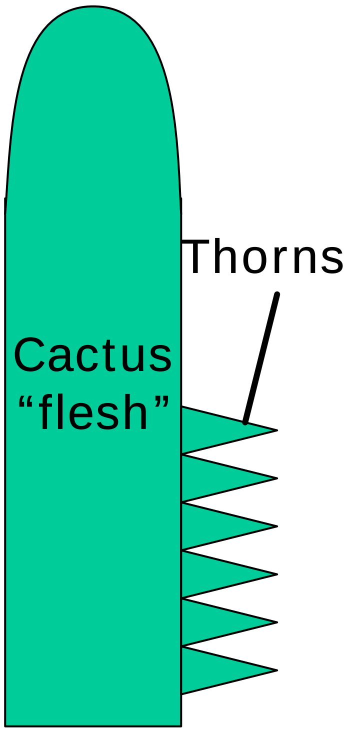
- A complete implementation of the Message Passing Interface (MPI) for heterogeneous, wide area environments
  - Based on the Argonne MPICH implementation of MPI (Gropp and Lusk)
- Requires services for authentication, resource allocation, executable staging, output, etc.
- Programs run in wide area without change
  - Modulo accommodating heterogeneous communication performance
- See also: MetaMPI, PACX, STAMPI, MAGPIE

www.globus.org/mpi

Cactus

(Allen, Dramlitsch, Seidel, Shalf, Radke)

- Modular, portable framework for parallel, multidimensional simulations
- Construct codes by linking
  - Small core (flesh): mgmt services
  - Selected modules (thorns): Numerical methods, grids & domain decomps, visualization and steering, etc.
- Custom linking/configuration tools
- Developed for astrophysics, but not astrophysics-specific



www.cactuscode.org

High-Throughput Computing and Condor

- High-throughput computing
  - CPU cycles/day (week, month, year?) under non-ideal circumstances
  - “How many times can I run simulation X in a month using all available machines?”
- Condor converts collections of distributively owned workstations and dedicated clusters into a distributed high-throughput computing facility
- Emphasis on policy management and reliability

www.cs.wisc.org/condor

the globus project
www.globus.org

Object-Based Approaches

- Grid-enabled CORBA
  – NASA Lewis, Rutgers, ANL, others
  – CORBA wrappers for Grid protocols
  – Some initial successes
- Legion
  – U.Virginia
  – Object models for Grid components (e.g., “vault”=storage, “host”=computer)

Portals

- N-tier architectures enabling thin clients, with middle tiers using Grid functions
  - Thin clients = web browsers
  - Middle tier = e.g. Java Server Pages, with Java CoG Kit, GPDK, GridPort utilities
  - Bottom tier = various Grid resources
- Numerous applications and projects, e.g.
  - Unicore, Gateway, Discover, Mississippi Computational Web Portal, NPACI Grid Port, Lattice Portal, Nimrod-G, Cactus, NASA IPG Launchpad, Grid Resource Broker, …

Common Toolkit Underneath

- Each of these programming environments should not have to implement the protocols and services from scratch!
- Rather, want to share common code that...
  - Implements core functionality
    - SDKs that can be used to construct a large variety of services and clients
    - Standard services that can be easily deployed
  - Is robust, well-architected, self-consistent
  - Is open source, with broad input
- Which leads us to the Globus Toolkit™...

the globus project™
www.globus.org

Introduction to the Globus Toolkit™

Globus Toolkit™

- A software toolkit addressing key technical problems in the development of Grid enabled tools, services, and applications
  - Offer a modular "bag of technologies"
  - Enable *incremental* development of grid-enabled tools and applications
  - Implement standard Grid protocols and APIs
  - Make available under liberal open source license

General Approach

- Define Grid protocols & APIs
  - Protocol-mediated access to remote resources
  - Integrate and extend existing standards
  - “On the Grid” = speak “Intergrid” protocols
- Develop a reference implementation
  - Open source Globus Toolkit
  - Client and server SDKs, services, tools, etc.
- Grid-enable wide variety of tools
  - Globus Toolkit, FTP, SSH, Condor, SRB, MPI, …
- Learn through deployment and applications

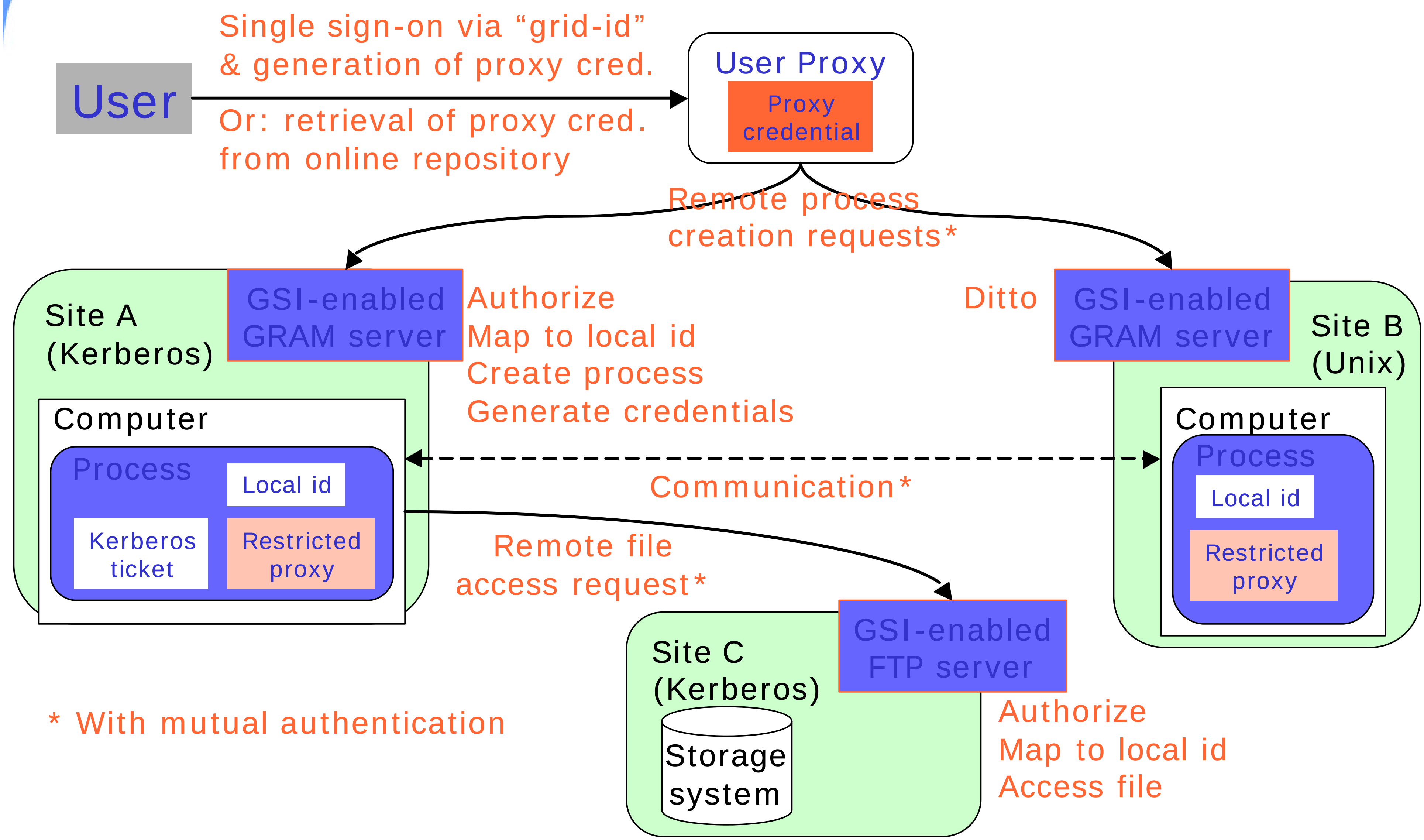
Key Protocols

- The Globus Toolkit™ centers around four key protocols
  - Connectivity layer:
    - *Security*: Grid Security Infrastructure (GSI)
  - Resource layer:
    - *Resource Management*: Grid Resource Allocation Management (GRAM)
    - *Information Services*: Grid Resource Information Protocol (GRIP)
    - *Data Transfer*: Grid File Transfer Protocol (GridFTP)
- Also key collective layer protocols
  - Info Services, Replica Management, etc.

Grid Security Infrastructure (GSI)

- Globus Toolkit implements GSI protocols and APIs, to address Grid security needs
- GSI protocols extends standard public key protocols
  – Standards: X.509 & SSL/TLS
  – Extensions: X.509 Proxy Certificates & Delegation
- GSI extends standard GSS-API

GSI in Action
"Create Processes at A and B that Communicate & Access Files at C"

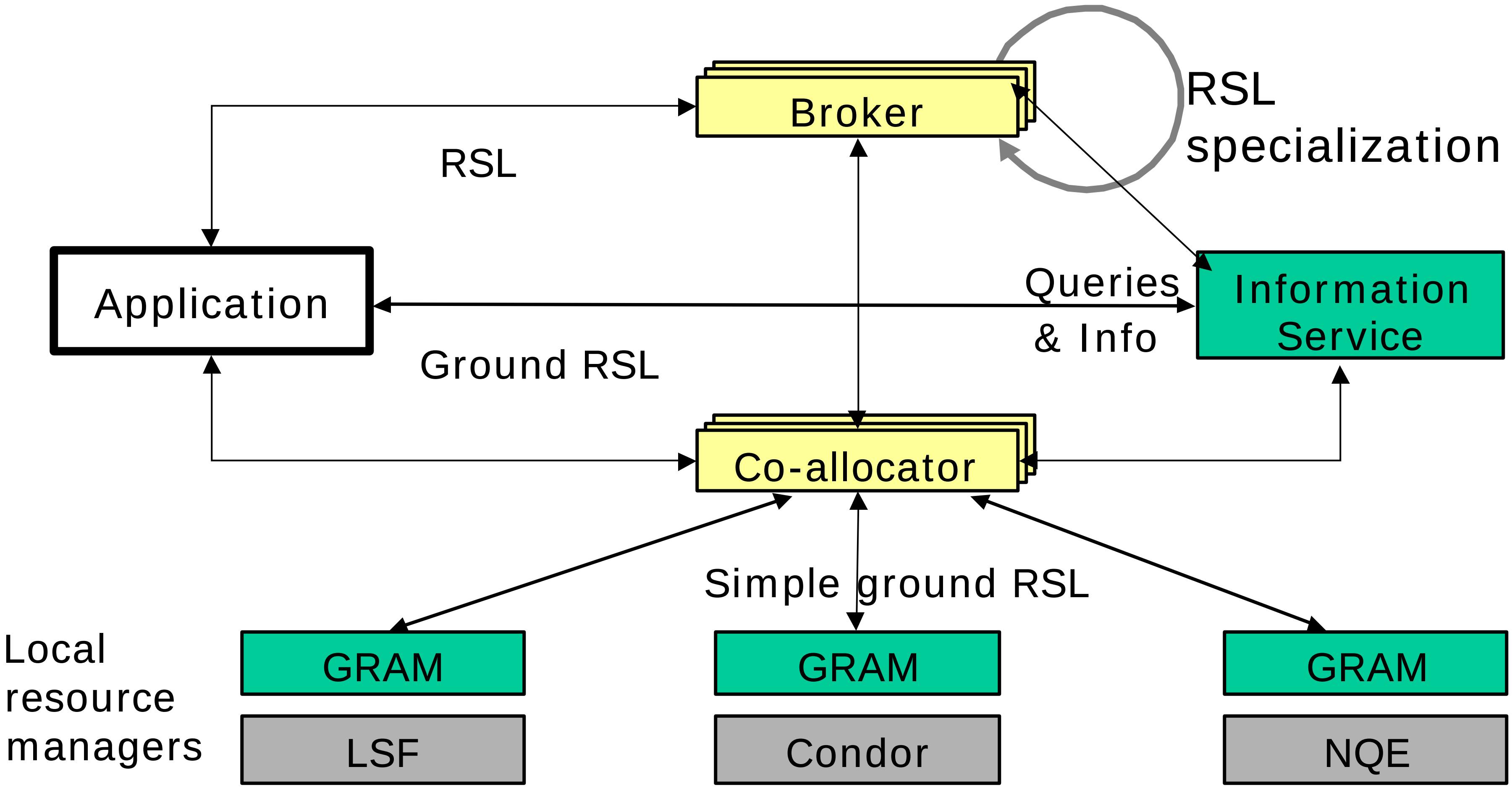


* With mutual authentication

Resource Management

- The Grid Resource Allocation Management (GRAM) protocol and client API allows programs to be started and managed on remote resources, despite local heterogeneity
- Resource Specification Language (RSL) is used to communicate requirements
- A layered architecture allows application-specific resource brokers and co-allocators to be defined in terms of GRAM services
  – Integrated with Condor, PBS, MPICH-G2, …

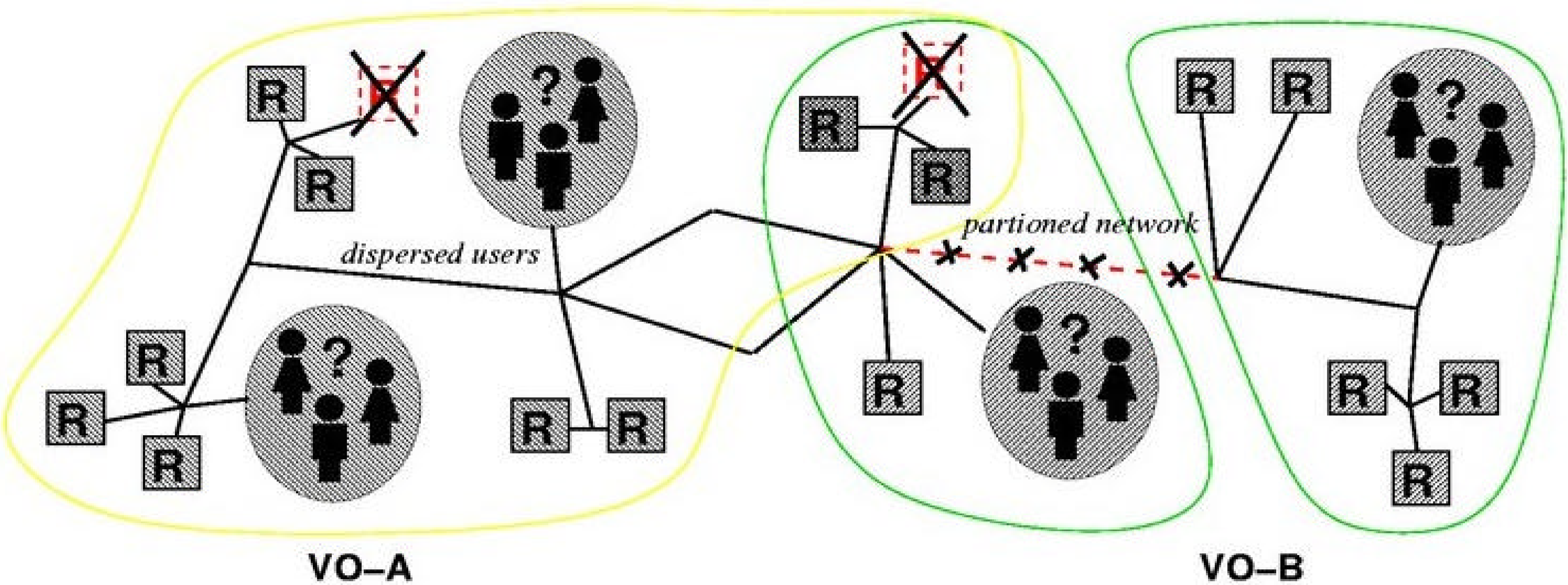
Resource Management Architecture



Data Access & Transfer

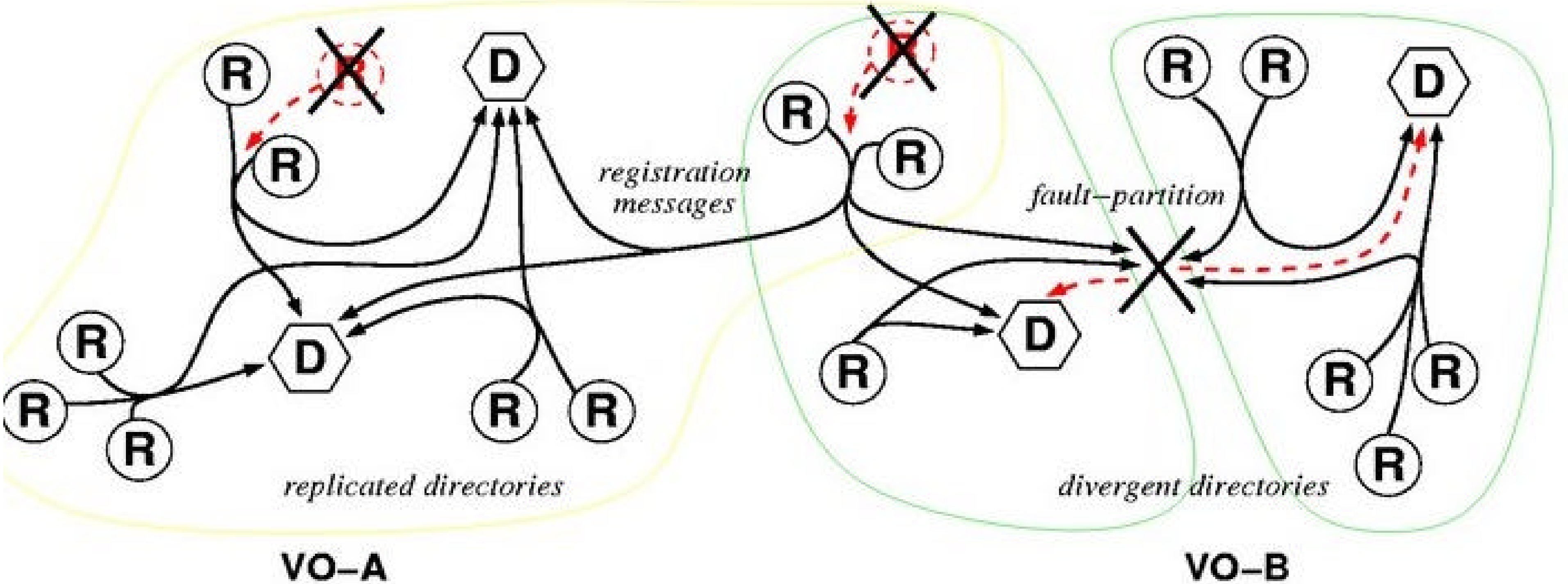
- GridFTP: extended version of popular FTP protocol for Grid data access and transfer
- Secure, efficient, reliable, flexible, extensible, parallel, concurrent, e.g.:
  – Third-party data transfers, partial file transfers
  – Parallelism, striping (e.g., on PVFS)
  – Reliable, recoverable data transfers
- Reference implementations
  – Existing clients and servers: wuftpd, ncftp
  – Flexible, extensible libraries in Globus Toolkit

The Grid Information Problem



- Large numbers of distributed “sensors” with different properties
- Need for different “views” of this information, depending on community membership, security constraints, intended purpose, sensor type

The Globus Toolkit Solution: MDS-2



Registration & enquiry protocols, information models, query languages

– Provides standard interfaces to sensors
– Supports different “directory” structures supporting various discovery/access strategies

Summary

- The Grid problem: Resource sharing & coordinated problem solving in dynamic, multi-institutional virtual organizations
- Grid architecture emphasizes *systems problem*
  - Protocols & services, to facilitate interoperability and shared infrastructure services
- Globus Toolkit™: APIs, SDKs, and tools which implement Grid protocols & services
  - Provides basic software infrastructure for suite of tools addressing the *programming problem*